How to tame runaway SQL statements

Abakus
As na disku.

dejavu Data at your service.

Oracle Optimizer

- 5-table join:
  - Join Order
    - = 1*2*3*4*5 = 120 possible permutations.
  - Join method = NL/HJ/MJ
    - = 3*3*3*3 = 81 (81*120 ~ 10.000)
  - Data Access = FTS/IUqS/I(Rg)S/ISkipS/IFFS/...
    - = 5*5*5*5*5 = 3.000 (*10.000)
    - = 30.000.000 possible access path
- 6-Table join is 2.733.750.000 possible access paths ...
- Limited number of gueses (2000).
- *https://chandlerdba.com/presentations/why-has-my-plan-changed-top-7-plan-stability-pitfalls-and-how-to-avoid-them/*

Oracle Optimizer

- 5-table join:
  - Join Order
    -
  - Jo
    -
  - Da
    -
    -
- 6-Tabl ns
  ...
- Limited number of gueses (2000).

Probability

- **0,000066667** - 5 table join
- **0,000000732** - 6 table join
- **0,000064135** - loto (12 numbers)

ENGLISH IDIOM:

A needle in a haystack

something that is very difficult to find

OysterEnglish.com

Oracle Optimizer

- Optimizer ALWAYS get it wrong (for complex sql).
- Sometimes it changes its mind and usually that makes only a minor difference.
- Sometimes is not a minor difference.
- Sometimes is not good at all.
- Sometimes is very very bad.

- https://chandlerdba.com/presentations/why-has-my-plan-changed-top-7-plan-stability-pitfalls-and-how-to-avoid-them/

Make IT

SIOUG 2023

How to tame runaway SQL statements

Boris Oblak
Abakus plus d.o.o.

ORACLE CERTIFIED PROFESSIONAL

ORACLE Gold Partner
Specialized
Oracle Database

AS Abakus As na disku

dejavu Data at your service.

Abakus Plus d.o.o.

- History
  - From 1992
  - ~20 employees
- DBA Applications
  - DejaVu
  - ARBITER
  - APPM
- Enterprise Applications
  - Document Management
  - Newspaper Distribution
  - Flight Information System
- Services
  - OS & Network admin
  - DBA, Programming
- Infrastructure
  - > 20 years of experience with

    High Availability on GNU/Linux
- Hardware
  - Servers, SAN, ceph,
  - Firewalls,
  - Backup Server

Abakus and Oracle

- Oracle database on linux
  - Abakus: 1995
    (Oracle 7.1.5, Forms 3.0)
  - Oracle: 1997
- Parallel execution
  - Abakus: 2004
    (SIOUG 2004: Vzporedno Izvajanje operacij s PL/SQL – Boris Oblak)
  - Oracle: 2007
    dbms_parallel_execute

APPM

Abakus Plus Performance Monitor

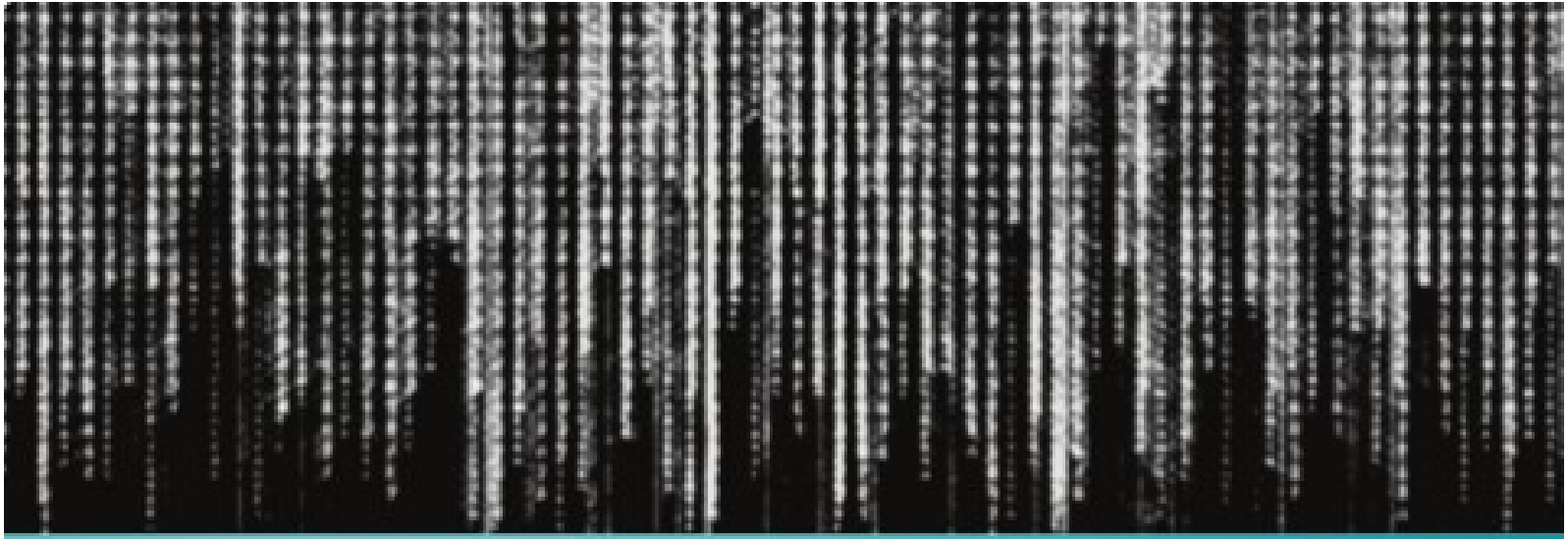
- For Oracle Database Standard Edition
- Made by DBAs for DBAs
- Temporal performance comparison
- Resource allocation optimization
- Database performance tracking
- Performance bottleneck optimization

www.abakus.si

Backup server

supports Oracle Databases and OLVM VMs

- **Backup**
  takes no time
- **Recovery**
  data recovery is almost instant
- **Disk space**
  backed up data takes up minimal amount of disk space
- **Availibility**
  data is always available and always in view
- **Security**
  backed up data can not be deleted without support personnel intervention
- **Alternative uses**
  BI analysis / reporting / DB upgrade verification / R&D testing / seamless business continuation



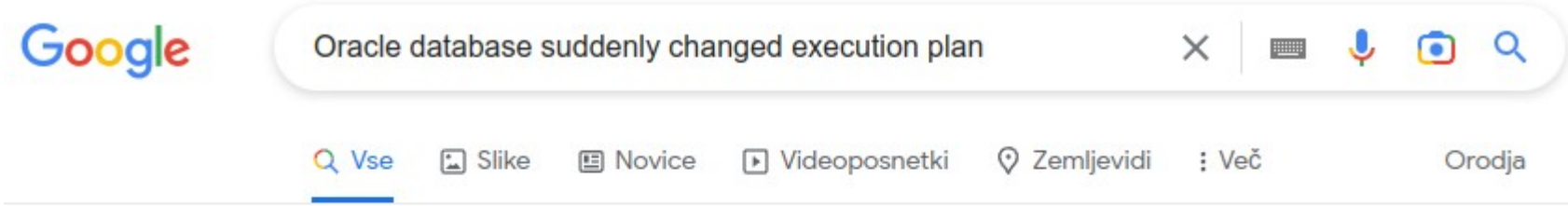
Deja Vu
virtual databases

Data at your service.

dejavu

References


Gorenjska Banka
GENERALI Zavarovalnica
Ljubljana Airport
EKDIS
Ekspresno. Ekonomično.
REPUBLIKA SLOVENIJA
MINISTRSTVO ZA OBRAMBO
NOVA BANKA
MILENIJUM OSIGURANJE
KONTROLA ZRAČNEGA PROMETA SLOVENIJE
Iskra
Hidria
Mestna občina Ljubljana
skbbanka
otp group
triglav
ANDRITZ
jata emona
LJUBLJANA
UNIVERZITETNA PSIHIATRIČNA KLINIKA LJUBLJANA
University Psychiatric Clinic Ljubljana
BANKA SLOVENIJE
SAVARe
MERKUR
TRELLEBORG
Trelleborg Slovenija, d.o.o.
SODO
SISTEMSKI OPERATER DISTRIBUCIJSKEGA OMREŽJA Z ELEKTRIČNO ENERGIJO
NLB Vita
Življenjska zavarovalnica
PRVA
MAGNETIK d.o.o.
PEST MANAGEMENT SOLUTIONS
Mercator
ZAVOD ZA ŠPORT RS PLANICA
PH Primorska hranilnica
GOODYEAR DUNLOP
SAVA TIRES
MM KARTON
studio ritem
Blubit
TKO
TOVARNA KOVINSKE OPREME
SAVA
HOTELS & RESORTS
CENTROSINERGIJA
PANTEON GROUP
Lonia
PRONET
CHOOSE THE FUTURE
hit alpinea
Kranjska Gora
LASERLINE
ORACLE
ROS d.o.o.
INFOTRANS
PARK POSTOJNSKA JAMA
ADRIA ANKARAN
HOTEL & RESORT



Cardinality Feedback (11g)

- Used when you have:
  - no stats.
  - a complex predicate, where Oracle can't calculate the cardinality.
  - multiple conjunctive/disjunctive filters (AND and OR).
- compares Estimate row counts (from the stats) to Actual row counts.
- select count(*) from v$sql_shared_cursor where use_feedback_stats = 'Y';
  - 1st run: parse and gets plan A, but it gets marked as having poor cardinality;
  - 2nd run: hard parse again and get plan B, we stick to plan B;
  - aged out of shared pool and back to loop again .. back to plan A;
  - when the plan is aged out the shared pool, all of this information is lost.

Statistics Feedback (12c)

- Cardinality feedback renamed to Statistics feedback and improved.
- use case: same as Cardinality Feedback.
  - v$sql.is_reoptimizable = 'Y' (causes a hard parse and new child cursor).
- A SQL plan directive may also be created (default in 12.1); SQL Plan Directives are associated with columns not SQL's and so can affect every SQL.
  - select * from dba_sql_plan_directives;
- two types of directives:
  - DYNAMIC_SAMPLING.
  - DYNAMIC_SAMPLING_RESULT.
  - *https://www.slideshare.net/MauroPagano3/sql-plan-directives-explained*
  - *https://mauro-pagano.com/2016/11/28/something-new-about-sql-plan-directives-and-12-2/*

SQL Plan Directives

- Adaptive statistics may compromise system stability.
- Disable Adaptive Statistics in OLTP for better stability:
  - In 12.1: (DBA_SQL_PLAN_DIRECTIVES)
    - dbms_spd.alter_plan_directive(id, 'ENABLED', 'NO');
    - dbms_spd.alter_plan_directive(id, 'AUTO_DROP', 'NO');
  - >= 12.2
    - optimizer_adaptive_plans = true; #default
    - optimizer_adaptive_statistics = false; #default

Bind variables

- Save the parsing overhead.
- Not so good for skewed datasets.
- OLTP: yes.
- DW: Using literals could be a smarter choice.
- Use bind when appropriate.

Other reasons

- Statistics (histograms).
- Init parameters.
- Oracle Automatic Memory Management (AMM) (avoid it).
- Dropped/added/changed an index you weren't using (Oracle can use index stats).
- Patching (and enabling optimizer fixes).
- Adaptive cursor sharing.
  - v$sql: is_bind_sensitive, is_bind_aware, is_shareable
- Adaptive execution plan.
  - Optimizer can switch between NESTED LOOPS and HASH JOIN during execution.

Common cause

- New data is added to the database.
- New statistics are calculated.
- Plan is aged out the shared pool (or invalidated).
- Maybe the statistics remain the same, no new data is added, only time passes by.
- The plan can change when data is added and new statistics are gathered.
- The plan can change when no data is added and no new statistics are gathered.
- The execution time may significantly increase even if the "plan_hash_value" remains unchanged and the data is almost unchanged.

Changed Plan

- Why did optimizer change the plan?
- How can i make the Optimizer to make »right« decision?
- What mesures can I implement to minimize the likehood of changes the plan?

How can a DBA improve consistency?

- Oracle have introduced four primary mechanisms:
  - Stored outlines (deprecated in 12c).
  - SQL Profiles.
  - SQL Patches.
  - SQL Plan Management Baselines.

SQL Profiles

- Hints, attached to SQL statement.
- Generally OPT_ESTIMATE (to change cardinality of the join predicate or column correlation).
- SQL Profiles are more or less statistics.
- They go stale; not as easy to update as statistics.
- Require the SQL Tuning License.
- Not available in Standard Edition.

SQL Patches

- Inject hints into a SQL statement.
- 11g, 12.1:
  - sys.dbms_sqldiag_internal.i_create_patch.
  - hint_text is limited to 500 bytes.
- Do not require the SQL Tuning License.
- Available in SE.
- => 12.2: it's offical and not internal function:
  - dbms_sqldiag.create_sql_patch.
  - hint_text is CLOB.

SQL Patches (11g, 12.1)

```
--
-- create_sql_patch_11.sql [sql_id] [hint] [patch_name]
--
var l_sql_id varchar2(13);
var l_hint  varchar2(500);
var l_name varchar2(30);

DECLARE
   l_sql_text CLOB;
   l_signature       dba_sql_patches.name%TYPE;
BEGIN
   :l_sql_id := '&1.';
   :l_hint   := '&2.';
   :l_name   := '&3.';

   SELECT sql_fulltext INTO l_sql_text
     FROM (
        SELECT sql_fulltext
          FROM gv$sqlarea
         WHERE sql_id = :l_sql_id
     ) WHERE rownum = 1;

   sys.dbms_sqldiag_internal.i_create_patch(sql_text    => l_sql_text,
                                            hint_text   => :l_hint,
                                            name        => :l_name,
                                            description => 'sql_id="' || :l_sql_id || '"',
                                            category    => 'MANUAL_' || to_char(SYSDATE, 'yyyymmddhh24miss'));

   sys.dbms_sqldiag.alter_sql_patch(:l_name, 'STATUS', 'ENABLED');
   sys.dbms_sqldiag.alter_sql_patch(:l_name, 'CATEGORY', 'DEFAULT');

END;
/
```

SQL Patches (>= 12.2)

```
--
-- create_sql_patch.sql [sql_id] [hint] [patch_name]
--
var l_sql_id varchar2(13);
var l_hint  varchar2(512);
var l_name varchar2(30);

DECLARE
   l_signature       dba_sql_patches.name%TYPE;
   l_ret             VARCHAR2(128);
BEGIN
   :l_sql_id := '&1.';
   :l_hint   := '&2.';
   :l_name   := '&3.';

   l_ret := dbms_sqldiag.create_sql_patch(sql_id      => :l_sql_id,
                                hint_text   => :l_hint,
                                name        => :l_name,
                                description => 'sql_id="' || :l_sql_id || '"',
                                category    => 'MANUAL_' || to_char(SYSDATE, 'yyyymmddhh24miss'));

   dbms_output.put_line ('Created patch name: [' || l_ret || ']');

   sys.dbms_sqldiag.alter_sql_patch(:l_name, 'STATUS', 'ENABLED');
   sys.dbms_sqldiag.alter_sql_patch(:l_name, 'CATEGORY', 'DEFAULT');

END;
/
```

SQL Patches (example)

```
create table t as select * from all_objects;
set feedback ON SQL_ID

SQL> select /*+ ordered_predicates */ avg(object_id) from t where translate(to_char(sin(object_id)),'1','a') = 'a' and owner =
'TECAJ';

SQL_ID: a00jff7yjpvqf

SQL> set feedback OFF
SQL> select * from table(dbms_xplan.display_cursor(null,null,'+hint_report'));

Hint Report (identified by operation id / Query Block Name / Object Alias):
Total hints for statement: 1
--------------------------------------------------------------------------------

   1 -  SEL$1
           -  ordered_predicates

SQL> @create_sql_patch a00jff7yjpvqf IGNORE_OPTIM_EMBEDDED_HINTS SAME_PLAN_1

SQL> select /*+ ordered_predicates */ avg(object_id) from t where translate(to_char(sin(object_id)),'1','a') = 'a' and owner =
'TECAJ';

SQL> select * from table(dbms_xplan.display_cursor(null,null,'+hint_report'));

Hint Report (identified by operation id / Query Block Name / Object Alias):
Total hints for statement: 1 (U - Unused (1))
--------------------------------------------------------------------------------

   1 -  SEL$1
         U -  ordered_predicates / rejected by IGNORE_OPTIM_EMBEDDED_HINTS

Note
-----
   - SQL patch "SAME_PLAN_1" used for this statement
```

SQL Patches (usage)

- gather_plan_statistics.
- ignore_optim_embedded_hints.
- Parallel. (EE, brute force, CPU!)
- ordered_predicates.
- more than one hint:
  - copy/paste OUTLINE block.
  - limits: 11.2 and 12.1: 500 bytes.

SQL Patches

```
SQL> select avg(object_id) from t where translate(to_char(sin(object_id)),'1','a') = 'a' and owner = 'TECAJ';

SQL> select * from table(dbms_xplan.display_cursor(null,null,'+hint_report +outline'));

SQL_ID  7ppcxd6cfsfgy, child number 0
-------------------------------------
select avg(object_id) from t where translate(to_char(sin(object_id)),'1'
,'a') = 'a' and owner = 'TECAJ'

Plan hash value: 2966233522

-----------------------------------------------------------------------------
| Id  | Operation          | Name | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |      |       |       |   322 (100)|          |
|   1 |  SORT AGGREGATE    |      |     1 |    10 |            |          |
|*  2 |   TABLE ACCESS FULL| T    |    31 |   310 |   322   (1)| 00:00:01 |
-----------------------------------------------------------------------------

Outline Data
-------------

  /*+
      BEGIN_OUTLINE_DATA
      IGNORE_OPTIM_EMBEDDED_HINTS
      OPTIMIZER_FEATURES_ENABLE('19.1.0')
      DB_VERSION('19.1.0')
      ALL_ROWS
      OUTLINE_LEAF(@"SEL$1")
      FULL(@"SEL$1" "T"@"SEL$1")
      END_OUTLINE_DATA
  */

Predicate Information (identified by operation id):
---------------------------------------------------

   2 - filter(("OWNER"='TECAJ' AND TRANSLATE(TO_CHAR(SIN("OBJECT_ID")),'
              1','a')='a'))
```

How to find proper plan?

- SGA:
  - If the plans are still in SGA.
- AWR:
  - EE edition.
  - Not all plans are saved in AWR.
- Abakus APPM:
  - Almost all plans are captured and saved.
  - It also works in SE/SE2.
  - History is not saved in the production database.
  - *https://www.abakus.si/sl/produkti/programska_oprema/appm*

APPM – execution histogram

SQL Text | Execution Plans | Explain Plan | ASH Statistics

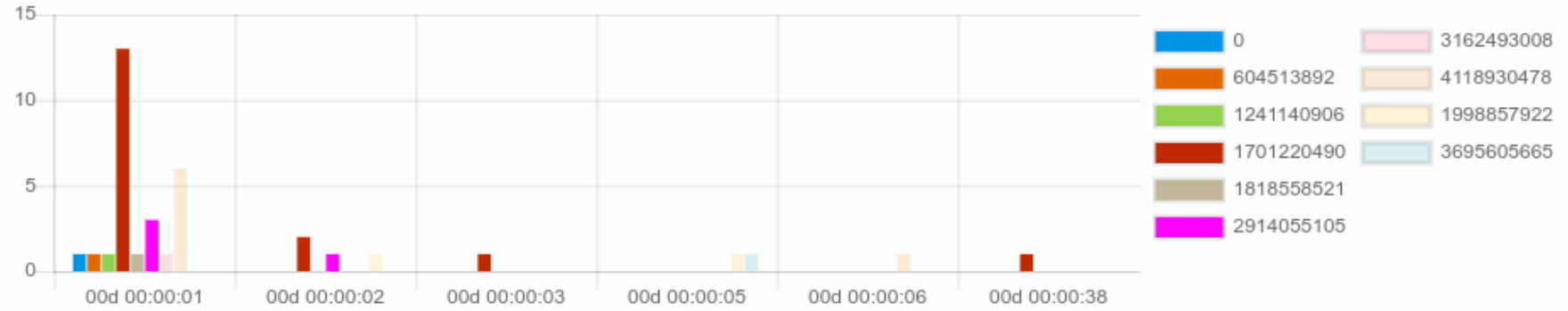
Compare Database: -- do not compare -- Group By: SQL ID

All Time ☑ Compare All Time ☐ Bucket Size: 1s

Date From: 2023-03-03 04:13 Compare From: 2023-03-03 04:13

Date To: 2023-03-03 05:13 Compare To: 2023-03-03 05:13

Run Report

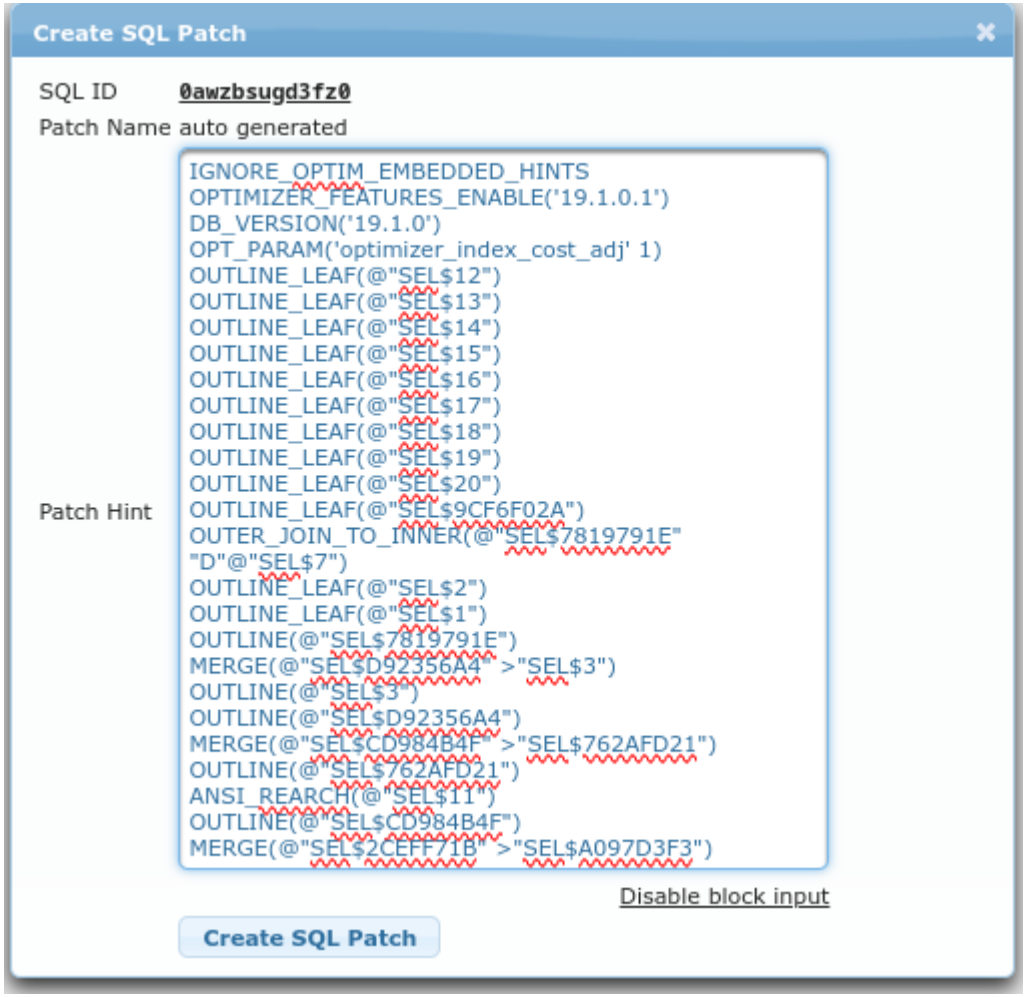


SQL ID	Plan Hash Value	Elapsed Time	Executions
0awzbsugd3fz0	0	00d 00:00:01	1
0awzbsugd3fz0	604513892	00d 00:00:01	1
0awzbsugd3fz0	1241140906	00d 00:00:01	1
0awzbsugd3fz0	1701220490	00d 00:00:01	13
0awzbsugd3fz0	1818558521	00d 00:00:01	1
0awzbsugd3fz0	2914055105	00d 00:00:01	3
0awzbsugd3fz0	3162493008	00d 00:00:01	1
0awzbsugd3fz0	4118930478	00d 00:00:01	6
0awzbsugd3fz0	1701220490	00d 00:00:02	2
0awzbsugd3fz0	1998857922	00d 00:00:02	1

APPM – copy Outline data

```
 32 - access("C"."CLIENT_ID"="U"."CLIENT_ID" AND "C"."EVENT_INDEX"="U"."EVENT_INDEX")
 34 - access("E"."EVENT_CODE"=:SYS_B_4)
 36 - access("R"."DOCUMENT_LOG_ID"="V"."R_DOCUMENT_LOG_ID")
 38 - access("T"."EVENT_CODE"="E"."EVENT_CODE" AND "T"."LANGUAGE_ID"='sl')

Outline Data:
-------------
IGNORE_OPTIM_EMBEDDED_HINTS
OPTIMIZER_FEATURES_ENABLE('19.1.0.1')
DB_VERSION('19.1.0')
OPT_PARAM('optimizer_index_cost_adj' 1)
OUTLINE_LEAF(@"SEL$12")
OUTLINE_LEAF(@"SEL$13")
OUTLINE_LEAF(@"SEL$14")
OUTLINE_LEAF(@"SEL$15")
OUTLINE_LEAF(@"SEL$16")
OUTLINE_LEAF(@"SEL$17")
OUTLINE_LEAF(@"SEL$18")
OUTLINE_LEAF(@"SEL$19")
OUTLINE_LEAF(@"SEL$20")
OUTLINE_LEAF(@"SEL$9CF6F02A")
OUTER_JOIN_TO_INNER(@"SEL$7819791E" "D"@"SEL$7")
OUTLINE_LEAF(@"SEL$2")
OUTLINE_LEAF(@"SEL$1")
OUTLINE(@"SEL$7819791E")
MERGE(@"SEL$D92356A4" >"SEL$3")
OUTLINE(@"SEL$3")
OUTLINE(@"SEL$D92356A4")
MERGE(@"SEL$CD984B4F" >"SEL$762AFD21")
OUTLINE(@"SEL$762AFD21")
ANSI_REARCH(@"SEL$11")
OUTLINE(@"SEL$CD984B4F")
MERGE(@"SEL$2CEFF71B" >"SEL$A097D3F3")
OUTLINE(@"SEL$11")
OUTLINE(@"SEL$A097D3F3")
ANSI_REARCH(@"SEL$0299E20D")
```

APPM – Create SQL Patch



SQL Baselines

- => 11g: SQL Plan Management (SPM). (EE)
  - Each SQL_ID can have more than one plan.
  - If better plan comes along, it can be captured; it will not be used unless we say that's OK.
  - Evolve plans (allow optimizer to select best plan from defined list).
- >= 18c: SPM is in Standard Edition.

SQL Baselines (SE2)

- SQL Plan Management does not require a license for Oracle Diagnostics Pack or Oracle Tuning Pack.
- SE2 and DBCS SE Summary: Only one SQL plan baseline per SQL statement is allowed and SQL plan evolution is disabled.
- SE2 and DBCS SE Details:
  - SQL plan baselines can be created or captured using the following methods:
    - Auto capture (OPTIMIZER_CAPTURE_SQL_PLAN_BASELINE=TRUE)
    - Manual loading from the cursor cache (DBMS_SPM.LOAD_PLANS_FROM CURSOR_CACHE)
    - Migration from stored outlines (DBMS_SPM.MIGRATE_STORED_OUTLINE)
    - Import using DBMS_SPM.UNPACK_STGTAB_BASELINE
- *https://docs.oracle.com/en/database/oracle/oracle-database/19/dblic/index.html*

SQL Baselines (SE2) (cont)

- All capture and creation methods store only one SQL plan baseline per SQL statement.
- SQL plan baselines can be exported and imported using DBMS_SPM.CREATE_STGTAB_BASELINE, DBMS_SPM.PACK_STGTAB_BASELINE, and DBMS_SPM.UNPACK_STGTAB_BASELINE.
- Unused SQL plan baselines are not auto-purged.
- Alternative SQL execution plans for SQL statements are not added to the SQL plan history.
- SQL plan baselines can be altered and dropped (DBMS_SPM.ALTER_SQL_PLAN_BASELINE and DBMS_SPM.DROP_SQL_PLAN_BASELINE).
- The following DBMS_SPM functions and procedures are not allowed: CONFIGURE, LOAD_PLANS_FROM_AWR, LOAD_PLANS_FROM_SQLSET, and all functions and procedures associated with SQL plan evolution.

SQL Baselines - capture

- alter system/session set optimizer_capture_sql_plan_baselines=true;
- dbms_spm.load_plans_from_cursor_cache
  - sql_id
  - plan_hash_value
  - ...
  - enabled ('YES', 'NO')

SQL Baselines - apply

- create hinted SQL and capture plan.
- apply plan from hinted SQL to original SQL:
  - dbms_spm.load_plans_from_cursor_cache
    - sql_id (hinted)
    - plan_hash_value (hinted)
    - sql_handle (original)

SQL Baselines – evolve (EE)

- optimizer_capture_sql_plan_baselines=true;
- dbms_spm.evolve_sql_plan_baseline:
  - [sql_handle] (NULL – all handles)
  - [plan_name] (NULL – all plan names)

SQL Baselines – transfer

- dbms_spm.create_stgtab_baseline.
- dbms_spm.pack_stgtab_baseline.
- transfer stage table to another database (data pump, ...).
- dbms_spm.unpack_stgtab_baseline.

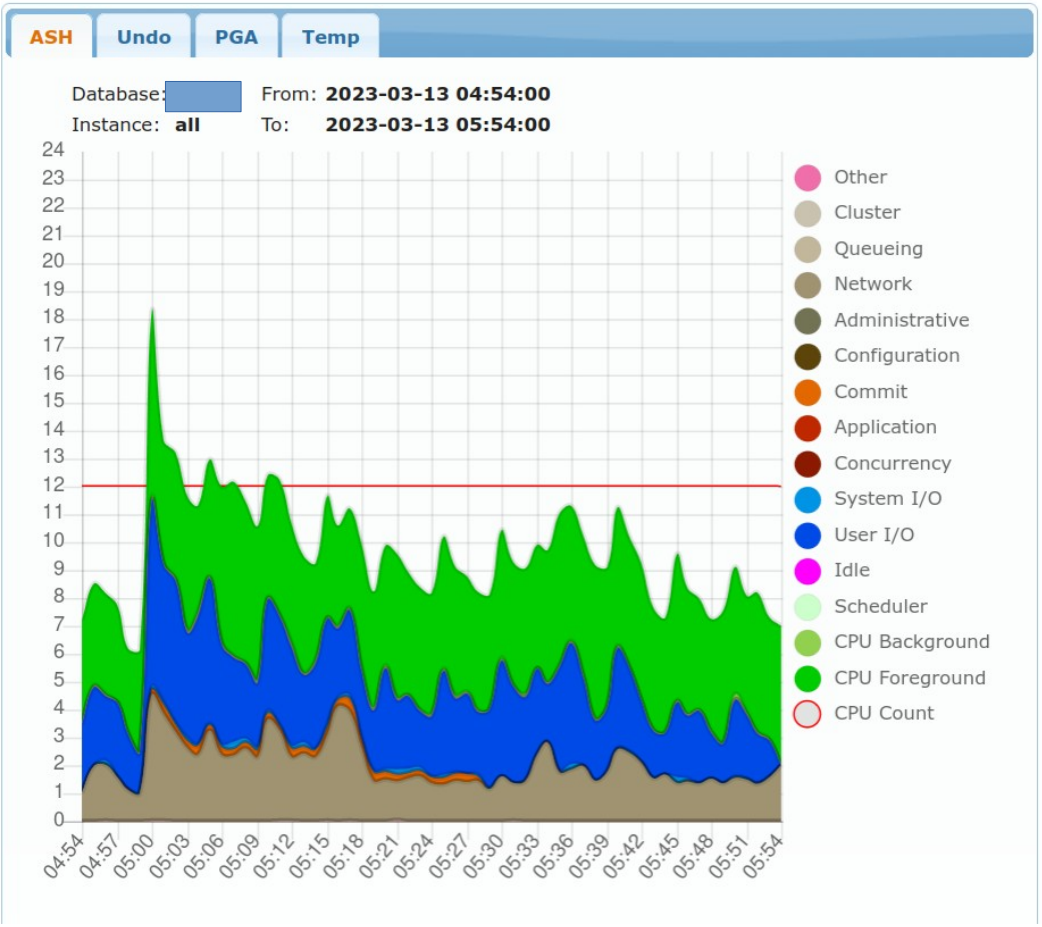
Performance regression (EE)

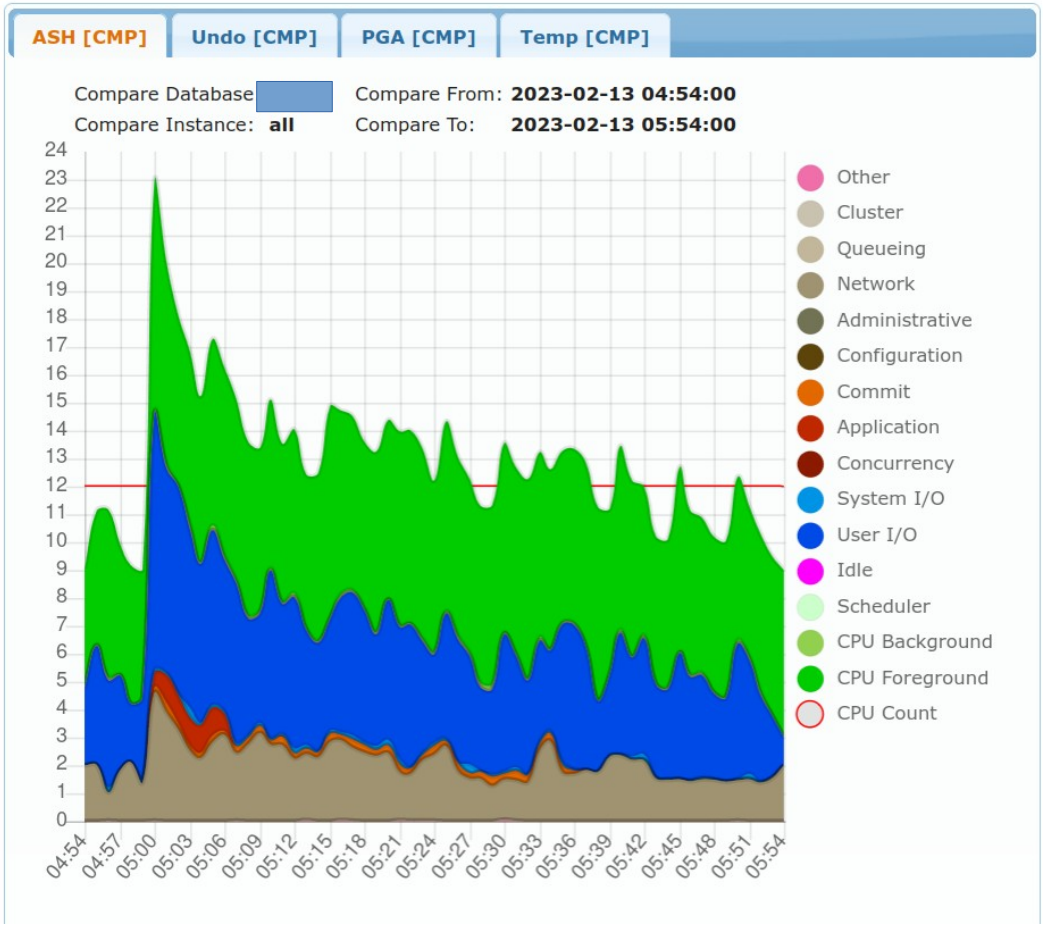
```
DECLARE
   l_tname   VARCHAR2(128);
   l_ename   VARCHAR2(128);
   l_cnt     PLS_INTEGER;
   l_handle  dba_sql_plan_baselines.sql_handle%TYPE;
BEGIN
   -- Create a SQL plan baseline for the problem query plan
   -- (in this case assuming that it is in the cursor cache)
   l_cnt := dbms_spm.load_plans_from_cursor_cache(sql_id          => '<problem_SQL_ID>'
                                                 ,plan_hash_value => <problem_plan_hash_value>
                                                 ,enabled         => 'no');
   SELECT p.sql_handle INTO l_handle
     FROM v$sqlarea s
     JOIN dba_sql_plan_baselines p
       ON p.signature = s.exact_matching_signature
      AND rownum = 1;

   -- Set up evolve
   l_tname := dbms_spm.create_evolve_task(sql_handle => l_handle);
   dbms_spm.set_evolve_task_parameter(task_name => l_tname
                                     ,parameter => 'ALTERNATE_PLAN_SOURCE'
                                     ,VALUE     =>
                         'CURSOR_CACHE+AUTOMATIC_WORKLOAD_REPOSITOR+SQL_TUNING_SET');
   dbms_spm.set_evolve_task_parameter(task_name => l_tname
                                     ,parameter => 'ALTERNATE_PLAN_LIMIT'
                                     ,VALUE     => 'UNLIMITED');

   -- Evolve
   l_ename := dbms_spm.execute_evolve_task(l_tname);

   -- Optionally, choose to implement immediately
   l_cnt := dbms_spm.implement_evolve_task(l_ename);
END;
/
```

Standard Edition (18c+)

- Save (pack) baselines before upgrade.
- Restore (unpack) problematic SQL ID baseline.
- APPM perfomance monitor:
  - Find old (good) plan.
  - Create SQL Patch (copy/paste outline section).
  - Create baseline.
- APPM:
  - Compare pre and post upgrade statements.
  - Find statements, that are executing slower.
  - Create baselines.
  - *http://www.abakus.si/sl/produkti/programska_oprema/appm*

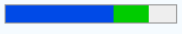
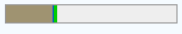
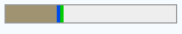
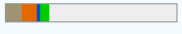
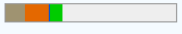
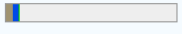
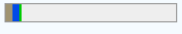
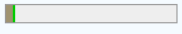
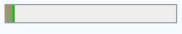
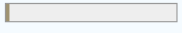
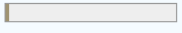
APPM – compare SQL





APPM Compare - SQL

Top Table (compare mode)

Activity	Activity [CMP]	SQL_ID (per exec)	SQL Text	Duration	Duration [CMP]	Absolute Diff	Ratio
		dfn0jgqac7rpr	SELECT POLICA ,STEVILKAFAKTURE ,OPISPRODUKTA ,DATU	00d 00:59:54.00	00d 00:59:42.00	00d 00:00:12	+00.33%
		9w9mnf0wa74xq	select O.AGENT, O.COMMISSION_AMOUNT, O.POLICY, O.P	00d 00:40:56.00	00d 00:50:41.00	00d 00:09:45	-19.24%
		64jzsru70cuyw	INSERT INTO CUSTOMERS_IMPORT_INTERFACE (CUSTOMER_	00d 00:35:38.00	00d 00:16:33.00	00d 00:19:05	+53.55%
		94fugqnz8uz62	INSERT INTO CUSTOMERS_IMPORT_INTERFACE (CUSTOMER_S	00d 00:09:03.50	00d 00:10:20.00	00d 00:01:16	-12.34%
		7t1yvzxjz63zf	SELECT ID_IZRACUN_EZP GFNR FROM OFFERS O WHERE OFF	00d 00:08:55.00	00d 00:07:44.00	00d 00:01:11	+13.27%
		akasjb37u7tdb	DECLARE job BINARY_INTEGER := :job; next_date TIM	00d 00:07:47.00	00d 00:09:44.00	00d 00:01:57	-20.03%
		7j766wj9nhntc	INSERT INTO MEDALLIA_ALL_DATA (ID, TOUCHPOINT_NUMB	00d 00:07:21.00	00d 00:08:19.00	00d 00:00:58	-11.62%
		cbac0jwx2yvqt	INSERT INTO MEDALLIA_ALL_DATA (ID, TOUCHPOINT_NUMB	00d 00:04:40.00	00d 00:05:57.00	00d 00:01:17	-21.57%
		4cu5cwqfbm27n	SELECT SUM(AX_NTPRPS) PREMIJA, AX_VTNRPS POLICA, A	00d 00:03:17.00	00d 00:03:34.00	00d 00:00:17	-07.94%
		g8bua48dxc3tq	INSERT /*+ BYPASS_RECURSIVE_CHECK */ INTO "PROVIS"	00d 00:02:53.00	00d 00:02:44.00	00d 00:00:09	+05.20%
		2cm5f6kanphf7	INSERT INTO MEDALLIA_ALL_DATA (ID, TOUCHPOINT_NUMB	00d 00:02:23.00	00d 00:04:45.00	00d 00:02:22	-49.82%
		37trttjatmzs6	SELECT P.ROWID AS POR, I.FIRM, P.POLICY, DECODE (I	00d 00:01:46.00	00d 00:01:40.00	00d 00:00:06	+05.66%
		ajsaadxxrjh01	INSERT INTO MEDALLIA_ALL_DATA (ID, TOUCHPOINT_NUMB	00d 00:01:37.50	00d 00:02:31.50	00d 00:00:54	-35.64%
			UPDATE POLICIES P SET P.OFFER_NUMBER =				

APPM Compare - SQL

SQL Statement

Instance: #1 exa302vm1.sias.si/ljprod (LOCAL)

SQL ID: bnz9cahq44dmk

Child Number: 0

Refresh | Pin | Unpin | Flush | Kept versions: 0

SQL Text | Other Children | Execution Plan | SQL Baselines | SQL Patches | Statistics

Plan Hash / Child	Last Active	Executions	**Elapsed	**Rows	**Fetches	**Buffer Gets	SQL Patch	SQL Baseline	Opts
3625424013 / 0 @ 1	2023-03-13 06:44:53	1	00:00:17	15	2	69			Create Baseline

*** Columns marked with asterisk are **average** numbers, based on number of executions.*

APPM Compare - SQL

SQL Statement

Instance: #1 exa302vm1.sias.si/ljprod (LOCAL)

SQL ID: bnz9cahq44dmk

Child Number: 0

Refresh | Pin | Unpin | Flush | Kept versions: 0

SQL Text | Other Children | Execution Plan | SQL Baselines | SQL Patches | Statistics

Plan Hash / Child	Last Active	Executions	**Elapsed	**Rows	**Fetches	**Buffer Gets	SQL Patch	SQL Baseline	Opts
3625424013 / 0 @ 1	2023-03-13 06:44:53	1	00:00:17	15	2	69			Create Baseline

** *Columns marked with asterisk are **average** numbers,*

Database upgrade

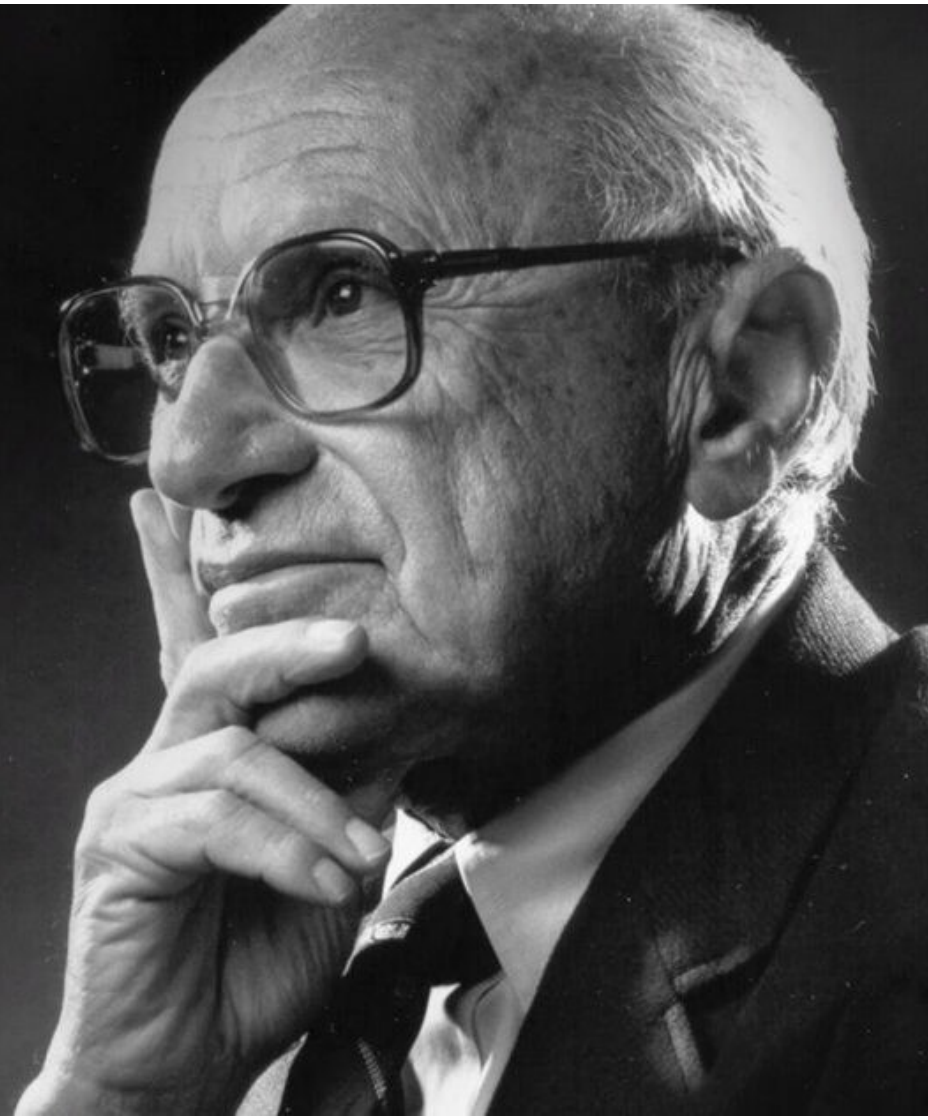
- Gather all baselines.
- Upgrade the database.
- All SQL statements will run as in the old databse.
- In case of EE, plan regression will be performed.

Database upgrade

- Gather all baselines.
- Upgrade the database.
- All SQL statements will run as in the old databse.
- In case of EE, plan regression will be performed.
- Is this really a good idea?

A Milton Friedman favorite political aphorism:

"There's no such thing as a free lunch."



There's no such thing as a free lounch

- Parsing overhead (more baselines, more overhead).
- SE2:
  - One baseline per sql_id.
  - SQL plan evolution is disabled.
    - problem: different plans depending on bind variables.

Recommendations

- Upgrade:
  - pre upgrade: load and pack baselines (failsafe).
- Load/enable only nesessary/critical SQL statements baselines.
- Upgrade:
  - post upgrade: unpack baselines for problematic SQL statements (statements, whose execution time is prolonged).
- Packaged applications:
  - Baselines for critical SQL statements should be part of the installation procedure.

ORA-03113: end-of-file on communication channel

Boris Oblak
Abakus plus d.o.o.
boris.oblak@abakus.si